M&G No. 50037.67US01

INSERTING DEVICE SPECIFIC CONTENT

Field of the Invention

The present invention relates generally to server-hosted application software, and more particularly to altering display properties of objects in a server-hosted application environment.

Background of the Invention

Many Internet-capable devices are available to consumers. Specifically, the number of mobile Internet-capable devices is increasing rapidly. Some Internet-capable devices integrate several functions. For example, a personal digital assistant (PDA), or a cellular telephone may be used for Internet access in addition to other shared functions. A PDA or a cell phone usually has a very different Internet capability than a desktop computer, based upon limitations of screen size, connectivity and system resources. For example, a PDA display may not display very large fonts, pictures, movies, or tables of data or long text strings that would be viewable on a full-size display. It is desirable to accommodate as many devices as possible for Internet content to maximize the benefit of web-based services.

Accompanying the increase in mobile devices accessing the Internet is the number of services that provide active content to the user. For example, a mobile device user-interface (UI), such as an ASP.NET page, may be constructed of a hierarchy of server-side controls. When a page is executed, the objects in the page hierarchy are processed to render the content that is transmitted to the client device. During the rendering process, formatting and layout information specified for the controls is transformed into markup tags interpreted by the client device to produce the desired appearance on the display. When specifying web-application UI for mobile devices, the application author often needs to tailor elements of the UI for different devices or browsers. For example, a label identifying a mail message may be long and descriptive on a device with a larger screen, and short on a device with a smaller screen. Generally, device-specific markup requires the developer to utilize conditional

constructs or to use stylesheets or includes. However, since such device customizations are relatively minor, they have high maintainability costs for tasks such as interspersing code or managing multiple files for small visual element differences.

Summary of the Invention

Briefly stated, a server-based application includes at least one page file that identifies one or more server objects. The page file describes the particular layout and interaction of the server objects, such as controls, in such a way that content may be transmitted to a device requesting the page file. When designing the page file, a developer may use a declarative statement so that certain properties of the specified controls may be altered based on characteristics of the target device.

In an aspect of the invention, a server receives an instruction to transmit content to a target device (e.g., a PDA, cellphone, other mobile device, personal computer, internet-enabled television, or the like). The instruction may be a request generated externally, such as from the device itself, or it may be an instruction generated locally, such as from an application on the server (e.g., time-based content generation software). The content may be a web page, for example. The instruction to transmit the content may include data identifying the target device. A runtime process provides a response to the request for content based upon characteristics of or the type of target device.

In another aspect of the invention, a server contains a page file describing the layout and properties of the content. Controls in the page file define the size, shape and textual properties of the content. A runtime process renders content for a specific target device based upon the page file. When compiled in a runtime process to create a class, the declarative statements in the page file generate code to set control properties for the content based upon the destination device type. Once a page file is compiled for a specific target device, an instance of the compiled class may be instantiated without further compiles for every request from a device of the requesting type. The present invention provides a system and method for inserting device-specific content into a server application for improved runtime device-specific content delivery.

Brief Description of the Drawings

FIGURE 1 is a functional block diagram illustrating an exemplary environment for practicing the invention.

FIGURE 2 shows an exemplary server computer that illustrates an operational environment for implementing the invention.

FIGURE 3 shows a functional block diagram of server components in an illustrative environment for applying a declarative construct for device-specific content rendering.

FIGURE 4 is a functional diagram illustrating an illustrative control hierarchy to override control properties in an example of the invention.

FIGURE 5 is a flow diagram illustrating a process for declaratively altering properties of server objects used in a server-based application, in accordance with an embodiment of the invention.

Detailed Description of the Invention

The present invention now will be described more fully with reference to the accompanying drawings, which form a part hereof, and which show, by way of illustration, specific exemplary embodiments by which the invention may be practiced. This invention may, however, be embodied in many different forms and should not be construed as limited to the embodiments set forth herein; rather, these embodiments are provided so that this disclosure will be thorough and complete, and will fully convey the scope of the invention to those skilled in the art. Like numbers refer to like elements throughout. As will be appreciated by one of skill in the art, the present invention may be embodied as methods or devices. Accordingly, the present invention may take the form of an entirely hardware embodiment, an entirely software embodiment or an embodiment combining software and hardware aspects. The following detailed description is, therefore, not to be taken in a limiting sense.

Computer storage media may include volatile and nonvolatile, removable and non-removable media implemented in any method or technology for storage of information, such as computer readable instructions, data structures, program modules, or other data. Examples of computer storage media include RAM, ROM,

EEPROM, flash memory or other memory technology, CD-ROM, digital versatile disks (DVD) or other optical storage, magnetic cassettes, magnetic tape, magnetic disk storage or other magnetic storage devices, or any other medium which can be used to store the desired information and which can be accessed by a computing device.

Briefly stated, the present invention is directed towards providing a system and method for optimizing markup in a just-in-time compiling environment for such devices as mobile devices. Among other things, disclosed is a system that employs a declarative construct to provide control overrides for device-specific content rendering. First, an illustrative operating environment and computing server will be described. Next, components used to implement declarative constructs for device-specific content rendering are described. Finally, methods for implementing declarative constructs will be disclosed.

Illustrative Operating Environment

FIGURE 1 is a functional block diagram illustrating an exemplary environment for practicing the invention. Figure 1 includes a server 102, a network 110, a software development environment 103 and network capable devices 116_A, 116_B and 116_C. Server 102 includes server objects 106, mobile Internet component 302, and page file 108.

Mobile devices 116_{A-C}, are products such as cellular telephones, pagers, hand-held electronic devices, programmable and non-programmable consumer electronics, personal computers, PDAs, and watches, for example. Although described here in the context of mobile devices, it will be appreciated that the teachings of the invention have equal applicability to many other types of target devices, such as personal computers, internet-enabled television sets, and the like. The focus on mobile devices in this disclosure is for simplicity of discussion only.

Mobile devices 116_{A-C}, may have a variety of capabilities and features. For example, a cell phone may have a numeric keypad and a few lines of monochrome LCD display on which only text may be displayed. A POCKET PC may have a touch sensitive screen, a stylus, and several lines of color LCD display in which both text and graphics may be displayed. A computer may have a keyboard, mouse, speakers, microphone, and a relatively large area on which to display forms.

Network 110 connects mobile Internet component 302 with mobile devices 116_{A-C}. Communication between mobile Internet component 302 and mobile devices 116_{A-C} through network 110 generally includes a request, which happens to be an HTTP request 120 in this instance, and a response, which happens to be an HTTP response 121 in this instance. Generally, the HTTP request 120 includes device identification data that identifies properties of the particular type of target device. For example, if mobile device 116_C issues the HTTP request 120, the device identification data may identify mobile device 116_C as a WAP-enabled cellular phone. As discussed more fully later, this information may be utilized to provide specific content to the target device. Alternatively, the request 120 may take the form of an instruction generated locally by an application on the server 102 programmed to transmit information to a device without a request having been issued by the device. In one example, an application on the server 102 may be configured to cause content to be delivered to a paging unit periodically without a request issued by the paging unit.

Network 110 may employ any form of computer readable media for communicating information from one electronic device to another. Also, network 110 can include the Internet in addition to local area networks (LANs), wide area networks (WANs), direct connections, such as through a universal serial bus (USB) port, other forms of computer-readable media, or any combination thereof. On an interconnected set of LANs, including those based on differing architectures and protocols, a router acts as a link between LANs, enabling messages to be sent from one to another. Also, communication links within LANs typically include twisted wire pair or coaxial cable, while communication links between networks may utilize analog telephone lines, full or fractional dedicated digital lines including T1, T2, T3, and T4, Integrated Services Digital Networks (ISDNs), Digital Subscriber Lines (DSLs), wireless links including satellite links, or other communications links known to those skilled in the art. Furthermore, remote computers and other related electronic devices may be remotely connected to either LANs or WANs via a modem and temporary telephone link. In essence, network 110 includes any communication method by which information may travel from any of mobile devices 116_{A-C} to server 102.

Server 102 is an example of a computing device, and is described in more detail in conjunction with FIGURE 2. Server 102 stores, retrieves, and executes applications and/or objects in response to requests for information from mobile devices 116_{A-C}. Server 102 executes mobile Internet component 302 in generating a device-specific response, such as HTTP response 121, that is transmitted through network 110 to requesting mobile devices 116_{A-C}. Server 102 may also contain other application programs and components and may be used for a variety of purposes related or unrelated to the present invention.

Software development environment 103 provides a software developer the tools for developing server-based applications, such as server-based application 107, for server 102. Software development environment 103 may be as simple as a text editor used with a file transport protocol (FTP) application for transmitting and receiving programs to and from server 102, or it may include a suite of software development tools such as one or more compilers, debuggers, source code control applications, and team development tools, for example. Such software development tools typically make the software applications easier to develop, debug, and maintain.

Mobile Internet component 302 provides device identification data regarding the capabilities of the target device, such as mobile devices 116_{A-C}. Identification data may be extracted from the HTTP request 120 or may be supplied by a local application. Mobile Internet component 302 utilizes server objects 106 and page file 108 created by a software development environment 103 to build the server-based application 107 and provide optimized content, according to one embodiment of the invention. Mobile Internet component 302 executes processes based on the capabilities of a target device that transform pages, forms, controls, and the like into content suitable for viewing on mobile devices 116_{A-C}.

The server-based application 107, such as may be created by the software development environment 103, typically includes pages (e.g., page file 108) and other server objects 106, such as forms and active controls. The pages and server objects operate in cooperation to execute as an application on the server 102 and provide feedback and an interface to the target device over the network 110.

Server objects 106 include active controls, forms, and other objects (collectively referred to sometimes as "controls") for performing specialized tasks on behalf of the server-based application 107, such as displaying information to and receiving input from users. Controls encapsulate data that dictates how a control should be rendered for use by a mobile device. Controls are employed to display text and images on a mobile device 116_{A-C}. A control may be used, for example, for collecting address information. The control may display address fields, prompt a user for address and name information, and validate inputted information. Other examples of server objects 106 may be a radio button control for receiving a selection, a free text control for receiving textual input, or an OK or CANCEL button. Another type of server object is a form control. Form controls may be considered as container controls with no visible representation, employed to call render methods of each of the controls contained in the form. Although controls like Form may not have a visible representation, they may (and often do) emit their own markup.

Importantly, each server object 106 has properties that affect how the server object 106 is displayed on a target device 116_{A-C}. For instance, a free text control will likely have a property that defines the font size of the text displayed by the control. Likewise, a button control may have a property that defines the text displayed within the button. It will be appreciated that different mobile devices may have different display characteristics, which may impact how the server objects 106 should be rendered on those different devices. For instance, a cellular phone is likely to have a smaller display screen than a handheld personal computer.

The page file 108 includes information, such as a set of tags that correspond to server controls that govern the information that is presented to the target device. More particularly, the page file 108 includes many instructions that define which server objects 106 are used when rendering the response to the incoming request 120. In addition, and in accordance with the invention, the page file 108 includes additional declarative statements that may set the properties of those server objects 106 in accordance with the particular type of mobile device that issued the request. For instance, the page file 108 may include the following sample pseudo-code

to identify which server objects 106 are used to construct the web page being requested, and to alter properties of those server objects 106 based on the type of target device:

```
<mobile:Image runat=server id="image1" ImageUrl="myimage.gif">
 <DeviceSpecific>
      <Choice Device= "IsBlackAndWhite" ImageUrl="myimage_bw.gif"/>
      <Choice Device= "IsGrayscale" ImageUrl="myimage_gs.gif"/>
      <Choice Text= "My Image"/>
 </DeviceSpecific>
</mobile:Image>
```

In the above pseudo-code, the statement "mobile:Image" identifies the particular server object as an Image control, which is used to display a particular image on the target device. The statement "runat=server" indicates that the control will execute at the server 102. The statement "ImageUrl='myimage.gif" indicates that an ImageUrl property of the Image control is set to "myimage.gif" (the particular image to be displayed). In accordance with the invention, a DeviceSpecific tag is included within the Image tag to indicate that properties of the Image control may be overridden when the page is compiled by the mobile Internet component 302. In the above example, two Choice statements alternatively override the "ImageUrl" property based on whether the target device supports grayscale images or only black and white images. The "Device=" condition specifies a filter against which the target devices are evaluated. Another Choice statement sets the "Text" property of the control to "My Image" independent of the type of target device. Note that a Choice statement without a Device= condition operates as a choice that matches any device (essentially the default choice).

In short, the server-based application 107 includes at least one page file 108 that identifies one or more server objects 106. The page file 108 describes the particular layout and interaction of the server objects 106 in such a way that a page of information may be transmitted to a target device. When designing the page file 108, a developer may use the inventive DeviceSpecific construct, introduced above and

detailed below, so that certain properties of the specified controls may be altered based on the particular type of target device. Details of this operation are provided below.

Illustrative Server Environment

FIGURE 2 shows an exemplary server computer that illustrates an operational environment for implementing the invention. Figure 2 includes a server 102 as in Figure 1. Server 102 includes a central processing unit 212, a video display adapter 214, an input/output interface 224, a network interface unit 210, and a mass memory, all in communication with each other via a system bus 222. The mass memory generally includes RAM 216, ROM 232, and one or more permanent mass storage devices, such as hard disk drive 228, optical drive 226 and optionally a tape drive, and/or floppy disk drives (not shown). The mass memory stores an operating system 220 for controlling the operation of server 102. Basic input/output system ("BIOS") 218 is provided for controlling the low-level operation of server 102.

Mass memory also stores WWW server 230 and the mobile Internet component 302. WWW server 230 is a web server application that hosts network connections between users of a network and the server 102. Mobile Internet component 302 and WWW server 230 include computer executable instructions which, when executed, generate displays and perform the logical functions. Mass memory may also include additional application programs 250.

As illustrated in FIGURE 2, server 102 is operative to communicate with the Internet, or some other communications network, via network interface unit 210, which is constructed for use with various communication protocols including the TCP/IP protocol. A bidirectional communication 211 to network interface unit 210 is illustrated in Figure 2. Server 102 may optionally include an SMTP handler application for transmitting and receiving e-mail, an HTTP handler application for receiving and handing HTTP requests, and an HTTPS handler application for handling secure connections. An HTTPS handler application may initiate communication with an external application in a secure fashion.

When a device such as mobile device 116_A, shown in FIGURE 1, requests display content from server 102, the request is routed to the WWW server 230.

When the request is for server pages, the WWW server 230 forwards the request to mobile Internet component 302.

FIGURE 3 shows a functional block diagram of server components in an illustrative environment for applying a declarative construct for device-specific content rendering. FIGURE 3 shows a server 102, as in FIGURE 1, including mobile Internet component 302, page file 108, server objects 106, network interface 210, and bidirectional communication 211. Mobile Internet component 302 further includes a mobile runtime process 312 and a control hierarchy 304. Control hierarchy 304 is described below in greater detail in conjunction with FIGURE 4. Briefly described, the control hierarchy 304 is a working copy or class of a compiled version of the server objects 106 implicated by the requested page file 108.

Mobile runtime process 312 receives requests, responses, and/or information from network interface 210. In essence, the mobile runtime process 312, in response to a request for a particular page, builds the control hierarchy 304 for a target device from the requested page (e.g., page file 108) in conjunction with the server objects 106. To facilitate that end, the mobile runtime process 312 includes device capabilities component 303 and rendering component 308.

The device capabilities component 303 is programmed to receive information from an incoming page request (HTTP request 120 in FIGURE 1) and identify the particular type of target device. Different devices may have different capabilities as discussed in conjunction with FIGURE 1. Device capabilities component 104 may include a database of "known" devices, it may query a device in real-time for capabilities, or it may determine capabilities from additional information sent by the device. For example, a device may include information about screen size within a header in its request sent to the server. Device capabilities component 104 may determine that the device capabilities are unknown. In such a case, device capabilities component 104 may send a default set of capabilities, an error, or some other message so indicating.

The rendering component 308 is used to create actual response content for transmission to the target device. In other words, once the server-based application has been compiled and content for the requested page has been constructed, the actual

content returned is prepared by the rendering component 308 for transmission to the target device.

Network interface 210 transmits and receives bidirectional communication 211. Such communication may be transmitted and received using protocols including hypertext transport protocol (HTTP), transmission control protocol/Internet protocol (TCP/IP), ftp, email, direct file transfer, combinations thereof, and the like. In essence, any transmission protocol capable for transmitting information over a network may be used in conjunction with network interface 210 to send information to and receive information from devices.

Some embodiments of mobile Internet component 302, its subcomponents, and other components have been described above. In light of this disclosure, it will be understood that components and interactions of the components within server 102 could be changed, added, or removed without departing from the spirit and scope of this invention.

FIGURE 4 is a functional diagram illustrating an illustrative control hierarchy to override control properties in an example of the invention. A control hierarchy, such as control hierarchy 304, may be used in the process of compiling a page file 108 into a server-based application 107. The control hierarchy 304 may be thought of as an object class that defines the executable portion of the server-based application 107. A typical control hierarchy 304 includes an object hierarchy or tree structure including the controls or server objects identified by the requested page file. The control hierarchy may include parent controls, such as Image control 401, and one or more child controls, such as DeviceSpecific control 403, thus, providing a hierarchical structure. Control hierarchy 304 may also include other controls 410.

In one embodiment, the control hierarchy 304 is created based on a declarative construct used when developing the page file 108. The declarative construct was introduced above in conjunction with FIGURE 1, and generally takes the form of the following markup tag:

```
<DeviceSpecific>
    <Choice Device="_____" Property= argument>
    <Choice Device="_____" Property= argument>
    <Choice Device="_____" Property= argument>
</DeviceSpecific>
```

In this embodiment, the DeviceSpecific tag is nested within a tag that identifies a particular control. The existence of the DeviceSpecific tag indicates to the process compiling the requested page that the particular control has properties that may be overridden based on different types of devices. The Choice tag includes a condition that identifies the particular type of device (e.g., the Device="__" statement) and the value to apply to a particular property if that condition is true (e.g., the "Property= argument" statement).

In operation, the compiling process (e.g., mobile runtime process 312) may walk the requested page file and create an object within the control hierarchy 304 for each identified control. A nested tag may indicate to the compiling process to create a child object. Thus, in accordance with the disclosed embodiment, the DeviceSpecific tag causes a child object to be created in the control hierarchy 304 under the particular control within which the DeviceSpecific tag exists. It is possible for a control to have multiple DeviceSpecific tags where each DeviceSpecific tag would be applied to the control independently. Each set of property overrides may be based on different device decisions than another set, further easing management of control property overrides. To illustrate the point more clearly, consider once again the sample pseudo-code introduced above in conjunction with the page file 108 of FIGURE 1:

```
<mobile:Image runat=server id="image1" ImageUrl="myimage.gif">
  <DeviceSpecific>
      <Choice Device= "IsBlackAndWhite" ImageUrl="myimage_bw.gif"/>
      <Choice Device= "IsGrayscale" Image Url="myimage_gs.gif"/>
      <Choice Text= "My Image"/>
  </DeviceSpecific>
</mobile:Image>
```

In this case, the control hierarchy 304 includes an Image control 401, which has a DeviceSpecific child control 403. In addition, the DeviceSpecific control 403 includes child objects 406-407 that each represents one of the alternative Choice tags above. Thus, the result of interpreting the page file 108 and building the control hierarchy 304 is a structure that identifies each of the controls (including their properties) that are used to create the display to be returned to the target device. Where appropriate, objects within the control hierarchy 304 include DeviceSpecific children that, when implemented, override certain properties of the parent control. The certain properties and their override values are individually stored within Choice objects, which are children of the DeviceSpecific object.

Operation of The Described Embodiments

FIGURE 5 is a flow diagram illustrating a process for declaratively altering properties of server objects used in a server-based application, in accordance with an embodiment of the invention. The process 600 begins at step 601 where a request is received from a target device for a page file. The request may be an HTTP request or other equivalent request for information over a network. At step 602, a determination is made whether a compiled version of the requested page already exists for the target device. Referring to the system described above, the device capabilities component 303 may be used to determine the particular type of target device, and the mobile runtime process 312 may then query whether a class exists to implement the requested page for that particular type of device.

If the determination at step 602 is negative, then, at step 603, the requested page is read. In accordance with this embodiment, the page has been developed using the DeviceSpecific construct described above for those controls or server objects for which device-specific properties may be appropriate. At step 604, a run-time process, such as mobile runtime process 312, compiles and builds an object class that describes the particular controls described within the requested page. In accordance with the invention, the object class may be a control hierarchy, such as that described above in conjunction with FIGURE 4, that includes objects for each control described within the requested page. Controls including DeviceSpecific information additionally include child controls that provide alternative values for particular

properties as may be set for different types of devices. In one particular embodiment, a "Choice" object contains a "property bag" that can hold an arbitrary set of properties. The "Choice" object is added to a "DeviceSpecific" object, which maintains an ordered set of "Choices." The "DeviceSpecific" object is added to the control, which maintains a reference to it. At step 606, the compiled class is stored.

When the compiled class is stored, or if the determination at step 602 was yes, then, at step 607, the object class is instantiated to service the incoming page request. Step 607 occurs for each new request for a page from a device matching the target device.

At step 608, execution of the requested page is begun. During page initialization, each control in the corresponding control hierarchy has an initialize method called on it. The initialize method determines if a "DeviceSpecific" object has been added to the control. If so, the initialize method requests the "DeviceSpecific" object to apply applicable property overrides for the control. The "DeviceSpecific" object goes through each Choice object in order, and requests the Choice object to evaluate itself against the device capabilities information (supplied by device capabilities component 303).

At step 610, the Choice objects are evaluated and, where appropriate, modify the corresponding properties of the parent control. The "Choice" object may use a "Device" property to evaluate itself. If a Choice object evaluates successfully, it then applies the corresponding property overrides. The Choice object would iterate over each property in its particular "property bag" and set the corresponding properties on the parent control accordingly. For instance, referring briefly to the control hierarchy 304 illustrated in FIGURE 4,.

At step 612, the device-specific page markup is rendered by a rendering subcomponent, such as rendering component 308 (FIGURE 3), and transmitted to the target device. For instance, once the server-based application has been initialized (as done at steps 608-610), the controls that form the application have been modified in accordance with the particular target device. Thus, when each control performs its respective portion of the execution of the application, the output from the control will

be tailored for the type of target device. The markup may then be returned to the target device in any conventional manner. The process 600 terminates at step 613.

In another embodiment, alternative code may be created to set the properties of the containing control directly by evaluating a series of conditions rather than creating child objects associated with each of the DeviceSpecific tags and Choice tags described in the page file.

For instance, at step 604 above, an object class may be constructed that includes a conditional construct representing each of the choices described in the page. One example of such alternative code may take the form:

```
Sub ApplyImageOverrides(image)
 If (Device = "IsBlackAndWhite") Then
  image.ImageUrl = "myimage_bw.gif"
 Else If (Device="IsGrayscale") Then
  image.ImageUrl = "myimage_gs.gif"
 Else
  image.Text = "My Image"
 End If
End Sub
```

and that code may be included in the object class generated when the page file is compiled. In this way, when the compiled page is instantiated, at step 607 above, the alternative code is executed, which evaluates the device and sets device-specific properties appropriately.

Each block of the flowchart illustrations, and combinations of blocks in the flowchart illustrations, can be implemented by computer program instructions. These program instructions may be provided to a processor to produce a machine, such that the instructions, which execute on the processor, create means for implementing the functions specified in the flowchart block or blocks. The computer program instructions may be executed by a processor to cause a series of operational steps to be performed by the processor to produce a computer implemented process such that the instructions which execute on the processor provide steps for implementing the functions specified in the flowchart block or blocks.

Accordingly, blocks of the flowchart illustration support combinations of means for performing the specified functions, combinations of steps for performing the specified functions and program instruction means for performing the specified functions. It will also be understood that each block of the flowchart illustration, and combinations of blocks in the flowchart illustration, can be implemented by special purpose hardware-based systems which perform the specified functions or steps, or combinations of special purpose hardware and computer instructions.

The above specification, examples, and data provide a complete description of the manufacture and use of the composition of the invention. Since many embodiments of the invention can be made without departing from the spirit and scope of the invention, the invention resides in the claims hereinafter appended.